New IP Timer Board

Support details

Tue, Jul 22, 1997

A new 8-channel IndustryPack timer board has been designed. An attractive feature of the board is that the logic in the gate array is automatically sourced from an inexpensive ROM chip when the board is powered up. This document explains how it is supported in IRM software.

The timer registers are accessed via the first 64 bytes of the IP I/O space assigned to the board. (For slot `c` on the 162 board, for example, this is `FFF58200`.) Setting a pulse delay or pulse width register is a simple memory write, so the analog control type can simply be a memory word type. There are 8 sets of four word-size registers used by each timer. Here is the layout for each timer, from T0–T7:

`delay` `delay_cntr` `width` `width_cntr`

The time units for these registers are specified by the prescale code in the timer's control register. The 16-bit unsigned delay and width registers are set by software. The corresponding read-only counter registers show the instantaneous counter values as a diagnostic.

Each timer has an associated control register. The 8 control registers are located following the 64 bytes of timer registers, say, for T0, at `FFF58240`, as follows:

Bit#	*Name*	
15–8	spare	
7	Enable timer output	
6	Prescale 1	0=> 0.1 μs, 1=> 1 μs, 2=> 10 μs, 3=> 100μs
5	Prescale 0	
4	External 10 MHz clock	
3	spare	
2	External trigger	
1	Synchronize	
0	Daisy chain enable (T1, T3, T5, T7 only)	

The contents of the control registers are set via analog channels whose analog control field targets it as a memory address. At reset time, they are refreshed to their last setting values. The daisy chain enable refers to triggering timer T(odd) from T(odd–1).

Clock event trigger selection/clearing

The board includes a 256-byte memory that is used to specify which Tevatron clock events are triggers for each timer. Each byte holds 8 bits, one bit per timer. The 256 bytes correspond to the 256 possible clock events. To select event 02 for timer 7, for example, the 3rd byte (00, 01, 02) in the array must have bit 7 set.

For each timer, a set of event selection channels is defined for the purpose of holding the events that can trigger that timer. The hardware, of course, can select to trigger a timer on up to 256 events, but in practice, such extensive multi-triggering is not needed. One may decide to define 8 channels, say, per timer, each of which can hold a triggering event# for that timer. (More such channels may be easily added later.) When the channel is set to a clock event, the corresponding bit in the array is checked. If it is set, then that clock event is presumably already used by another event selection channel to trigger that timer, so the setting is ignored. If the bit is clear, then it is set, and the clock event# is assigned as the setting value of that channel. If the channel is set to zero, the event selection bit is cleared.

The above scheme allows setting the trigger events, but we also need to verify that they are set, in order to furnish a reading value for such channels. A Data Access Table routine serves this purpose by checking the bit that corresponds to the given timer's trigger event# to be sure it is set. If it is not set, then the reading value is reset to zero, else it is set to the clock event#.

One would like additional assurance that the number of triggering clock events that are supposed to be in use for a given timer remains steady. To that end, another pseudo channel is defined for each timer channel whose reading reflects the total number of selected trigger events for that timer. If this event-clearing channel is set to zero, then all 256 clock event trigger bits are cleared for that timer. (If it is set to any nonzero value, the setting attempt is ignored.)

Auto-restore after reboot

When an IRM resets, all settings are restored to the hardware, since the reset could have occurred following a power outage. It is therefore important to have the above event-clearing channel occur before (*i.e.*, have a smaller channel# than) the set of event selection channels for that timer. This will cause all 256 bits to be at first cleared, then all selected clock event trigger bits to be set. The placement of the timer delay and width channels may be anywhere. But it may be best to place the channel that is used to hold the control register value *last* in the set of channels assigned to this timer board, so that all registers are loaded before the timers are enabled.

Analog control entry formats

As mentioned above, the delay and width registers are simply addressed as memory settings. But we also need to toggle the most significant bit to support a 16-bit unsigned range. As an example, the following would be the analog control field format for a T1 pulse width register in a timer board in slot `c` of a 162 board:

```
1 8   F 5   8 2 0 C
```

The 18 means a 16-bit memory write after toggling the ms bit. The address `F5820C` is automatically sign-extended to `FFF5820C`, because the high digit is `F`.

For the each of the event selection registers for Tx (T0–T7 range) on the same board, this format is used:

```
1 B   0 x   7 2 0 4
```

The base address for the memory on the slot `c` board is, by local convention, `72000000`. On this board, the 256-byte array is offset 256K bytes from the base address, so its base address is `72040000`. As only 16 bits are available in this formulation, the assumption is made that the array starts on a 64K boundary.

For the clear events channel for timer Tx, the following would be used:

```
1 B   1 x   7 2 0 4
```

For the control registers on this card, the following would work for T3:

```
0 A   F 5   8 2 4 6
```

Again, the control register channels should occur later than the other channels related to this board, so that all registers are loaded before the timers are enabled.

Data Access Table formats

For verifying the clock event trigger selections in the for the timers, the following DAT entry is used:

2 E	0 0	chan	—	—
—		—	—	count

Each channel in the range indicated is checked for having analog control type `1B`, signifying that it is a clock event selection channel. The hi-order word of the base address of the 256-byte array is taken from the last two bytes of this field. The timer#, which determines the bit# in each byte of the array that to be targeted, is taken from the second byte of the analog control field. Note that the event selection channels can be defined in a block that includes such channels for all timers on the IP board. One entry of this type can cover them all.

For the channels used for clearing the event select trigger bits for one timer, the following entry is used:

2 F	0 0	chan	ptr to event trig array	
—		max event#	timer#	count

Here, the pointer to the 256-byte array must be present in full, even though it must be on a 64K-byte boundary, given the limitations in space available in the analog control fields of the special event selection/clear channels. The max event# is included as an option, in case only low-numbered events are in use; it may save some processing time. The timer# indicates the first timer# (0–7 range) for which the tallying of event selection bits should be done. The count indicates for how many successive timers the tallying will be done.

For the delay and width channels, use the following entry:

3 0	0 0	chan	ptr to delay/width register	
—		—	step size	count

The memory will be read and the ms bit toggled before assigning the value into the data pool. This allows working with a full 16-bit positive range, if the full-scale and offset values are set properly in the analog descriptors.